1 SUMMARY

To **solve one or more sets of sparse Hermitian or complex symmetric linear unassembled finite-element equations, $\mathbf{AX}=\mathbf{B}$, by the frontal method,** optionally holding the matrix factor out-of-core in direct access files. Numerical pivoting is **not** performed so for Hermitian matrices it is primarily designed for the positive definite case. Use is made of high-level BLAS kernels. The coefficient matrix **A** must of the form

$$\mathbf{A}=\sum_{k=1}^{m}\mathbf{A}^{(k)}, \qquad (1)$$

with $\mathbf{A}^{(k)}$ nonzero only in those rows and columns that correspond to variables in the k-th element.

The frontal method is a variant of Gaussian elimination and involves the factorization

$$\mathbf{A}=\mathbf{PLD}(\mathbf{PL})^{\mathrm{H}} \text{ (Hermitian case), or } \mathbf{A}=\mathbf{PLD}(\mathbf{PL})^{\mathrm{T}} \text{ (symmetric case).}$$

where **P** is a permutation matrix, **D** is a diagonal matrix, and **L** is a unit lower triangular matrix. The solution process is completed by performing the forward elimination

$$(\mathbf{PL})\mathbf{DY}=\mathbf{B},$$

followed by the back substitution

$$(\mathbf{PL})^{\mathrm{H}}\mathbf{X}=\mathbf{Y} \text{ (Hermitian case) or } (\mathbf{PL})^{\mathrm{T}}\mathbf{X}=\mathbf{Y} \text{ (symmetric case).}$$

`ME62` stores the values of the entries in the factors and their indices separately. A principal feature of `ME62` is that, by holding the factors out-of-core, large problems can be solved using a predetermined and relatively small amount of in-core memory. At an intermediate stage of the solution, l say, the 'front' contains those variables associated with one or more of $\mathbf{A}^{(k)}$, $k=1,2,...,l$, which are also present in one or more of $\mathbf{A}^{(k)}$, $k=l+1,...,m$. For efficiency, the user should order the $\mathbf{A}^{(k)}$ so that the number of variables in the front (the 'front size') is small. For example, a very rectangular grid should be ordered pagewise parallel to the short side of the rectangle. The elements may be preordered using the HSL routine `MC63`.

`ME62` uses reverse communication.

ATTRIBUTES — Version: 1.0.1. (1 March 2008) **Types:** Real (single, double). **Remark:** Complex version of `MA62`. **Calls:** `_AXPY`, `_GERU`, `_GEMV`, `_TPSV`, `_TRSV`, `_GEMM`, `_TRSM`. **Helpful:** `MC63`. **Language:** Fortran 77. **Original date:** November 1999. **Origin:** I.S. Duff and J.A. Scott, Rutherford Appleton Laboratory.

2 HOW TO USE THE PACKAGE

2.1 Argument lists and calling sequences

There are six entries:

(a) The subroutine `ME62I/ID` must be called to initialize the parameters that control the execution of the package. This subroutine must be called once prior to calling other routines in the package.

(b) `ME62A/AD` must be called for each element to specify which variables are associated with it. This subroutine determines in which element each variable appears for the last time.

(c) `ME62J/JD` must be called for each element. This subroutine uses the information from `ME62A/AD` to determine the amount of real and integer storage required for the factorization.

(d) The use of `ME62P/PD` is optional. If direct access files are to be used, `ME62P/PD` must be called once prior to calling `ME62B/BD`.

(e) ME62B/BD must be called for each element to specify the nonzeros of $\mathbf{A}^{(k)}$ and, optionally, the corresponding element right-hand side(s) $\mathbf{B}^{(k)}$. ME62B/BD uses the information generated by ME62A/AD and ME62J/JD in the factorization of the matrix (1) and, if $\mathbf{B}^{(k)}$ are specified, ME62B/BD solves the equations $\mathbf{AX}=\mathbf{B}$ with right-hand side(s) $\mathbf{B}=\sum_{k=1}^{m}\mathbf{B}^{(k)}$.

(f) The use of ME62C/CD is optional. ME62C/CD uses the factor computed by ME62B/BD to solve for further right-hand sides. Several calls to ME62C/CD may follow a single call to ME62B/BD.

2.1.1 The initialization subroutine

To initialize control parameters, the user must make a single call of the following form:

The single precision version

```
CALL ME62I(ICNTL,CNTL,ISAVE)
```

The double precision version

```
CALL ME62ID(ICNTL,CNTL,ISAVE)
```

ICNTL is an INTEGER array of length 15 that need not be set by the user. This array is used to hold control parameters. On exit, ICNTL contains default values. If the user wishes to use values other than the defaults, the corresponding entries in ICNTL should be reset after the call to ME62I/ID. Details of the control parameters are given in Section 2.2.1.

CNTL is a REAL (DOUBLE PRECISION in the D version) array of length 5 that need not be set by the user. This array is used to hold control parameters. On exit, CNTL contains default values. If the user wishes to use values other than the defaults, the corresponding entries in CNTL should be reset after the call to ME62I/ID. Details of the control parameters are given in Section 2.2.1.

ISAVE is an INTEGER array of length 50 that need not be set by the user. This array is used to hold parameters that must be unchanged between calls to routines in the ME62 package.

2.1.2 Specification of which variables belong in each element

A call of the following form must be made for each element.

The single precision version

```
CALL ME62A(NVAR,IVAR,NDF,LAST,LENLST,ICNTL,ISAVE,INFO)
```

The double precision version

```
CALL ME62AD(NVAR,IVAR,NDF,LAST,LENLST,ICNTL,ISAVE,INFO)
```

NVAR is an INTEGER variable that must be set by the user to the number of variables in the element. This argument is not changed by the routine. **Restriction:** NVAR $\geq$ 1.

IVAR is an INTEGER array of length NVAR that must be set by the user to contain the indices of the variables associated with the element. These indices need not be in increasing order but must be distinct. This argument is not changed by the routine. **Restrictions:** 1 $\leq$ IVAR(I) $\leq$ LENLST and IVAR(I) $\neq$ IVAR(J), I, J = 1, 2, ..., NVAR.

NDF is an INTEGER variable that need not be set by the user. On each exit, it will be set to the largest integer so far used to index a variable. It must not be changed by the user between calls to ME62A/AD nor prior to subsequent calls to ME62J/JD and ME62B/BD. Note that, if the variables are not indexed contiguously, NDF will exceed the number of variables in the problem (see INFO(3) in Section 2.2.2).

LAST is an INTEGER array of length LENLST that need not be set by the user. On each exit from ME62A/AD, LAST(I) indicates the element in which the variable with index I last appeared or, if it has not appeared, LAST(I) is zero. On exit from the final call, if I has been used to index a variable, LAST(I) is the element at which variable I is fully summed and is zero otherwise. The first NDF entries of this array must not be changed between calls to ME62A/AD nor prior to subsequent calls to ME62J/JD and ME62B/BD.

LENLST is an INTEGER variable that must be set by the user to the dimension of array LAST. LENLST must be at least as large as the largest integer used to index a variable and must not be changed between calls to ME62A/AD. This argument is not changed by the routine. **Restriction:** LENLST ≥ 1.

ICNTL is an INTEGER array of length 15 that must be set by the user to hold control parameters. Default values are set by the call to ME62I/ID. Details of the control parameters are given in Section 2.2.1. Only ICNTL(I), I = 1, 2, and 8, are used by the routine. This argument is not changed by the routine.

ISAVE is an INTEGER array of length 50 that is used to hold parameters that must be unchanged between calls to routines in the ME62 package. This argument is changed by the routine.

INFO is an INTEGER array of length 20 that need not be set by the user. On each successful exit, INFO(1) is set to 0. Negative values of INFO(1) indicate a fatal error has been detected (see Section 2.3). If an error is detected, INFO(2) holds additional information concerning the error. INFO(I), I ≥ 3, are not accessed by the routine. This array must not be altered by the user.

2.1.3 Symbolic factorization of A

To determine the amount of storage required by the factorization, a call of the following form must be made for each element. The elements must have the same index lists and be in exactly the same order as when ME62A/AD was called. All the calls to ME62A/AD must be completed before ME62J/JD is called. Note that the storage is dependent on the control parameter ICNTL(5). If the user wishes to compute the storage required by different values of ICNTL(5), it is not necessary to recall ME62A/AD before repeating the sequence of calls to ME62J/JD.

The single precision version

```
CALL ME62J(NVAR,IVAR,NDF,LAST,ICNTL,ISAVE,INFO,RINFO)
```

The double precision version

```
CALL ME62JD(NVAR,IVAR,NDF,LAST,ICNTL,ISAVE,INFO,RINFO)
```

NVAR,IVAR are as in the corresponding calls to ME62A/AD but ME62J/JD does not check IVAR for duplicate indices. NVAR and IVAR are not changed by the routine.

NDF is an INTEGER variable which must be unchanged since the final call to ME62A/AD. This argument is not changed by the routine.

LAST is an INTEGER array of length NDF which must be unchanged since the final call to ME62A/AD. This argument is not changed by the routine.

ICNTL is an INTEGER array of length 15 that must be set by the user to hold control parameters. Default values are set by the call to ME62I/ID. Details of the control parameters are given in Section 2.2.1. Only ICNTL(I), I = 1, 2, 5, and 8, are used by the routine. This argument is not changed by the routine.

ISAVE is an INTEGER array of length 50 that is used to hold parameters that must be unchanged between calls to routines in the ME62 package. This argument is changed by the routine.

INFO is an INTEGER array of length 20 that need not be set by the user. On successful exit, INFO(1) is set to 0. Negative values of INFO(1) indicate a fatal error has been detected (see Section 2.3). If an error is detected, INFO(2) holds additional information concerning the error. On exit from the final call, INFO(I), I = 3, 4, 5, 6, contain information about the factorization and INFO(7) and INFO(8) are set to zero. Full details are given in Section 2.2.2. INFO(I), I ≥ 9, are not accessed by the routine. This array must not be altered by the user.

RINFO is a REAL (DOUBLE PRECISION in the D version) array of length 20 that need not be set by the user. On exit from the final call, RINFO(I), I = 1, 2 contain information about the symbolic factorization. Details are given in Section 2.2.2. RINFO(I), I $\geq$ 3, are not accessed by the routine. This array must not be altered by the user.

2.1.4 To set up direct access files

If the user wishes to keep in-core memory requirements low by using direct access files for the factors, a single call of the following form must be made. Note that for very large problems, using direct access files is recommended. If not, it may be necessary to use 64-bit integer arithmetic.

The single precision version

```
CALL ME62P(ISTRM,FILNAM,LENBUF,ICNTL,ISAVE,INFO)
```

The double precision version

```
CALL ME62PD(ISTRM,FILNAM,LENBUF,ICNTL,ISAVE,INFO)
```

ISTRM is an INTEGER array of length 2. ISTRM(1) and ISTRM(2) must be set by the user to specify the unit numbers of the direct access files for the reals in the factors and the indices of the variables in the factors, respectively. This argument is not changed by the routine. **Restrictions:** ISTRM(I) must lie in the range [1, 99], ISTRM(I) $\neq$ 6, ICNTL(1), or ICNTL(2) (I = 1, 2), and ISTRM(1) $\neq$ ISTRM(2).

FILNAM is a CHARACTER*128 array of length 2. If ICNTL(6) is reset by the user to a nonzero value, the user must set FILNAM(1) and FILNAM(2) to the filenames for the direct access files for the reals in the factors and the indices of the variables in the factors, respectively. If ICNTL(6) = 0 (the default), FILNAM is not accessed by the routine. This argument is not changed by the routine.

LENBUF is an INTEGER array of length 2. LENBUF(1) must be set by the user to the length, in COMPLEX (COMPLEX*16 in the D version) words, of the in-core buffer (workspace) associated with the direct access file for the reals in the factors (including the corresponding right-hand sides) and LENBUF(2) must be set by the user to the length, in INTEGER words, of the buffer associated with the direct access file for the indices of the variables in the factors. LENBUF(I) (I = 1, 2) have a crucial effect on the in-core memory requirements of ME62B/BD and ME62C/CD (see arguments LW and LIW in Sections 2.1.5 and 2.1.6). If NRHSB is the number of right-hand sides to be input to ME62B/BD, LENBUF should be chosen so that RINFO(1) + NDF*NRHSB = k_1*LENBUF(1) and RINFO(2) = k_2*LENBUF(2) with k_1, $k_2 \geq 1$ as small as available space permits (RINFO(1) and RINFO(2) as output from the final call ME62J/JD). LENBUF is not changed by the routine. **Restrictions:** LENBUF(I) > 0, I = 1, 2.

ICNTL is an INTEGER array of length 15 that must be set by the user to hold control parameters. Default values are set by the call to ME62I/ID. Details of the control parameters are given in Section 2.2.1. ICNTL(I), I = 1, 2, 3, 4, 6, and 8, are used by the routine. This argument is not changed by the routine. **Restrictions:** ICNTL(3) > 0 and ICNTL(4) > 0.

ISAVE is an INTEGER array of length 50 that is used to hold parameters that must be unchanged between calls to routines in the ME62 package. This argument is changed by the routine.

INFO is an INTEGER array of length 20 that need not be set by the user. On successful exit, INFO(1) is set to 0. Negative values of INFO(1) indicate a fatal error has been detected (see Section 2.3). If an error is detected, INFO(2) holds additional information concerning the error. INFO(I), I $\geq$ 3, are not accessed by the routine. This array must not be altered by the user.

2.1.5 To factorize A and optionally solve AX = B

A call of the following form must be made for each element. The elements must have the same index lists and be in exactly the same order as when ME62A/AD and ME62J/JD were called.

Note that all the calls to ME62J/JD for a particular problem must be completed before calling ME62B/BD.

The single precision version

```
    CALL ME62B(KIND,NVAR,IVAR,NDF,LAST,LAVAR,AVAR,NRHSB,RHS,LX,X,
   *           LENBUF,LW,W,LIW,IW,ICNTL,CNTL,ISAVE,INFO,RINFO)
```

The double precision version

```
    CALL ME62BD(KIND,NVAR,IVAR,NDF,LAST,LAVAR,AVAR,NRHSB,RHS,LX,X,
   *            LENBUF,LW,W,LIW,IW,ICNTL,CNTL,ISAVE,INFO,RINFO)
```

KIND is LOGICAL variable that must be set by the user to .TRUE. if the matrix **A** is Hermitian and to .FALSE. if **A** is complex symmetric. This argument must not be changed between calls to ME62B/BD and is not changed by the routine.

NVAR,IVAR,NDF,LAST are as in the corresponding calls to ME62J/JD. NVAR and NDF are not changed by the routine. On exit, the data in IVAR may have been permuted. Between calls to ME62B/BD, LAST is used as workspace and will be changed but on exit from the final call (or on an error return), LAST will have been restored to its original value.

LAVAR is an INTEGER variable that must be set by the user to the first dimension of the arrays AVAR and RHS. This argument is not changed by the routine. **Restriction:** LAVAR ≥ NVAR.

AVAR is a COMPLEX (COMPLEX*16 in the D version) array of dimensions LAVAR by NVAR. On entry, AVAR(I,J) must contain the contribution to entry (IVAR(I), IVAR(J)) in the matrix **A** from the current element (I, J = 1, 2,..., NVAR, J ≥ I). Contributions to the same entry from different elements are summed. This argument is changed by the routine.

NRHSB is an INTEGER variable that must be set by the user to the number of right-hand sides and must not be changed between calls to ME62B/BD. If the user does not wish to solve for any right-hand sides, NRHSB should be set to 0. This argument is not changed by the routine. **Restriction:** NRHSB ≥ 0.

RHS is a COMPLEX (COMPLEX*16 in the D version) array with leading dimension LAVAR. If NRHSB = 0, this array is not accessed. Otherwise on entry, the first NRHSB columns of RHS must be set by the user so that RHS(I,J) contains the contribution to component IVAR(I) of the J-th right-hand side from the current element (I = 1, 2,..., NVAR, J = 1, 2,..., NRHSB). Contributions to the same component from different elements are summed. This argument is changed by the routine.

LX is an INTEGER variable that must be set by the user to the first dimension of the array X. This argument is not changed by the routine. **Restriction:** If NRHSB ≥ 1, LX ≥ NDF.

X is a COMPLEX (COMPLEX*16 in the D version) array with leading dimension LX that need not be set by the user. If NRHSB = 0, this array is not accessed. Otherwise, the second dimension of X must be at least NRHSB and, on successful exit from the final call to ME62B/BD, if I has been used to index a variable, X(I,J) holds the solution for variable I to system J and is set to zero otherwise (I=1,2,...,NDF, J=1,2,..., NRHSB).

LENBUF is an INTEGER array of length 2. If the user is using direct access files, LENBUF must be unchanged since the call to ME62P/PD. Otherwise, LENBUF(1) must be set by the user to the length, in REAL (DOUBLE PRECISION in the D version) words, of the file for the reals in the factors (including the corresponding right-hand sides) and LENBUF(2) must be set by the user to the length, in INTEGER words, of the file for the indices of the variables in the factors. This array must not be changed between calls to ME62B/BD. This argument is not changed by the routine. **Restriction:** If direct access files are not being used, LENBUF(1) ≥ RINFO(1) + NDF*NRHSB, LENBUF(2) ≥ RINFO(2) (RINFO(1) and RINFO(2) as output from the last call to ME62J/JD).

LW is an INTEGER variable that must be set by the user to the dimension of array W. It must be unchanged between calls to ME62B/BD. This argument is not changed by the routine. **Restriction:** LW ≥ LENBUF(1) + INFO(6)*(NRHSB + INFO(6)) + 3 (INFO(6) as output from the last call to ME62J/JD).

W is a COMPLEX (COMPLEX*16 in the D version) array of length LW that is used as workspace by ME62B/BD. This array must be unchanged between calls to ME62B/BD. If direct access files are not being used (ME62P/PD not called), the first LENBUF(1) + 3 entries of W must be unchanged between the last call to ME62B/BD and any subsequent calls to ME62C/CD.

LIW is an INTEGER variable that must be set by the user to the dimension of array IW. It must not be changed between calls to ME62B/BD. This argument is not changed by the routine. **Restriction:** LIW ≥ LENBUF(2) + 3*INFO(6) (INFO(6) as output from the last call to ME62J/JD).

IW is an INTEGER array of length LIW that is used as workspace by ME62B/BD. This array must be unchanged between calls to ME62B/BD. If direct access files are not being used (ME62P/PD not called), the first LENBUF(2) entries of IW must be unchanged between the final call to ME62B/BD and any subsequent calls to ME62C/CD.

ICNTL is an INTEGER array of length 15 that must be set by the user to hold control parameters. Default values are set by the call to ME62I/ID. Details of the control parameters are given in Section 2.2.1. ICNTL(5) must be unchanged since calling ME62J/JD. ICNTL(I), I = 1, 2, 5, 7, 8, and 9 are used by the routine. This argument is not changed by the routine.

CNTL is a REAL (DOUBLE PRECISION in the D version) array of length 5 that must be set by the user to hold control parameters. Default values are set by the call to ME62I/ID. Details of the control parameters are given in Section 2.2.1. Only CNTL(1) is used by the routine. This argument is not changed by the routine.

ISAVE is an INTEGER array of length 50 that is used to hold parameters that must be unchanged between calls to routines in the ME62 package. This argument is changed by the routine.

INFO is an INTEGER array of length 20 that need not be set by the user. On successful exit, INFO(1) is set to 0. Negative values indicate a fatal error. For nonzero values of INFO(1), see Section 2.3. For details of the information contained in the other components of INFO, see Section 2.2.2. This array must not be altered by the user.

RINFO is a REAL (DOUBLE PRECISION in the D version) array of length 20. RINFO(1) and RINFO(2) must be unchanged since the final call to ME62J/JD. The other components need not be set by the user. On exit from the final call, RINFO holds information on the factorization. Full details are given in Section 2.2.2. This array must not be altered by the user.

2.1.6 To solve further systems AX = B

The single precision version

```
CALL ME62C(KIND,NRHSC,LX,X,LW,W,LIW,IW,ICNTL,ISAVE,INFO)
```

The double precision version

```
CALL ME62CD(KIND,NRHSC,LX,X,LW,W,LIW,IW,ICNTL,ISAVE,INFO)
```

KIND is LOGICAL variable that must unchanged since the calls to ME62B/BD. This argument is not changed by the routine.

NRHSC is an INTEGER variable that must be set by the user to the number of systems which are to be solved. This argument is not changed by the routine. **Restriction:** NRHSC ≥ 1.

LX is an INTEGER variable that must be set by the user to the first dimension of the array X. This argument is not changed by the routine. **Restriction:** LX ≥ NDF (NDF as output from the final call to ME62A/AD).

X is a COMPLEX (COMPLEX*16 in the D version) array of dimensions LX by NRHSC that must be set by the user so that if I has been used to index a variable, X(I,J) is the corresponding component of the right-hand side for the J-th system (J=1,2,..., NRHSC). On exit, if I has been used to index a variable, X(I,J) holds the solution for variable I to system J and is unchanged otherwise (J=1,2,..., NRHSC).

LW is an INTEGER variable that must be set by the user to the dimension of the array W. A sufficient value for LW is L1+L2, where L1 = LENBUF(1)+NRHSC*INFO(6). If direct access files are not being used (ME62P/PD was not called), L2 = 3, otherwise, L2 = INFO(5)*(INFO(6)+NRHSB). (INFO(5) and INFO(6) as output from the last call to ME62J/JD, and NRHSB as in the calls to ME62B/BD). This argument is not changed by the routine. **Restriction:** LW ≥ L1 + L2.

W is a COMPLEX (COMPLEX*16 in the D version) array of length LW. If direct access files are not being used (ME62P/PD was not called), the first LENBUF(1) + 3 entries of W must be unchanged since the last call to ME62B/BD and these entries are unchanged by ME62C/CD. Otherwise, W is used by ME62C/CD as workspace.

LIW is an INTEGER variable that must be set by the user to the dimension of the array IW. If direct access files are not being used (ME62P/PD was not called), LIW must be at least L1 = LENBUF(2). Otherwise, LIW must be at least L1 = LENBUF(2) + INFO(6) + 4. This argument is not changed by the routine. **Restriction:** LIW ≥ L1.

IW is an INTEGER array of length LIW. If direct access files are not being used (ME62P/PD was not called), the first LENBUF(2) entries of IW must be unchanged since the last call to ME62B/BD and these entries are unchanged by ME62C/CD. Otherwise, IW is used by ME62C/CD as workspace.

ICNTL is an INTEGER array of length 15 that must be set by the user to hold control parameters. Default values are set by the call to ME62I/ID. Details of the control parameters are given in Section 2.2.1. ICNTL(I), I = 1, 2, and 8, are used by the routine. This argument is not changed by the routine.

ISAVE is an INTEGER array of length 50 that is used to hold parameters that must be unchanged between calls to routines in the ME62 package. This argument is changed by the routine.

INFO is an INTEGER array of length 20 that need not be set by the user. On successful exit, INFO(1) is set to 0. Negative values of INFO(1) indicate a fatal error has been detected (see Section 2.3). If an error is detected, INFO(2) holds additional information concerning the error. INFO(I), I ≥ 3, are not accessed by the routine. This array must not be altered by the user.

2.2 Arrays for control and information

2.2.1 Control parameters

The elements of the arrays ICNTL and CNTL control the action of ME62A/AD, ME62J/JD, ME62P/PD, ME62B/BD, and ME62C/CD. Default values are set by ME62I/ID.

ICNTL(1) is the stream number for error messages and has the default value 6. Printing of error messages is suppressed if ICNTL(1) < 0.

ICNTL(2) is the stream number for warning messages and diagnostic printing. It has the default value 6. Printing of such messages is suppressed if ICNTL(2) < 0.

ICNTL(3) is the number of bytes for a complex word. ICNTL(3) has the default value 8 (16 for the D version).

ICNTL(4) is the number of bytes for an integer word. ICNTL(4) has the default value 4.

ICNTL(5) has the default value 16. ICNTL(5) controls the minimum number of variables which are eliminated at any one stage (except the last stage, when fewer than ICNTL(5) variables may remain). ICNTL(5) is only accessed on the first call to ME62J/JD and the first call to ME62B/BD. The value of ICNTL(5) on the first call to ME62B/BD should be the same as on the first call to ME62J/JD. Increasing ICNTL(5) in general increases the number of floating-point operations and real storage requirements but allows greater advantage to be taken of Level 3 BLAS.

ICNTL(6) has the default value 0. If it is reset to a nonzero value, the user must supply names for the direct access data sets in the parameter FILNAM when calling ME62P/PD.

ICNTL(7) is the block size for the numerical factorization of the frontal matrix. It controls the trade-off between Level 2 and Level 3 BLAS. If ICNTL(7) = 1, Level 2 BLAS is used to form the Schur complement. If ICNTL(7) ≥ 1, the Level 3 BLAS routine _GEMM is used with internal dimension ICNTL(7). Increasing ICNTL(7) increases the number of flops since symmetry is not exploited as well. The optimal value for ICNTL(7) depends on the computer being used. A value of ICNTL(7) less than one is treated as one and, if at some stage of the factorization, ICNTL(7) has a value which is larger than the current front size, ICNTL(7) is treated as the front size. Typical range: 16 to 64. Default value: 16.

ICNTL(8) is used to control the printing of error, warning, and diagnostic messages in ME62. It has default value 2. Possible values are:

0 No messages are output.

1 Only error messages are output.

2 Error and warning messages output.

3 As for 2, plus scalar parameters, arrays of length 2, and the control parameters on the first entry to ME62A/AD, ME62J/JD, ME62P/PD, and ME62B/BD, and scalar parameters on entry to ME62C/CD.

4 As for 3, plus INFO(I, (I = 1, 2,..., 8) on exit from final call to ME62J/JD, and the arrays INFO and RINFO on on exit from final call to ME62B/BD.

ICNTL(9) controls whether zeros within the frontal matrix are exploited. If ICNTL(9) = 0, the frontal matrix is treated as a dense matrix and zeros within the front are ignored. If ICNTL(9) is nonzero, the code will look for zeros occurring within the frontal matrix and will try to avoid unnecessary operations using zeros. This option can increase the amount of data movement but can also give worthwhile savings in the computation and in the number of entries in the factors if some variables are involved in only a few elements and these elements are well separated in the element order. The default value is 1.

ICNTL(10) controls whether a packed triangular form is used. If ICNTL(10) = 0 (the default value), a packed triangular form is used. In this case, the Level 2 BLAS routine _TPSV is used to perform triangular solves. If ICNTL(10) is reset to a nonzero value, a packed triangular form is not used. This increases the real storage required for the factors but the advantage is that, for multiple right-hand sides, the Level 3 BLAS routine _TRSM is used.

ICNTL(11) to ICNTL(15) are currently not used but are given a default value of zero and they may be used in a later release of the code.

CNTL(1) has the default value zero. If, during the factorization, the absolute value of any pivot is less than or equal to CNTL(1), the computation terminates (see INFO(1) = –12).

CNTL(2) to CNTL(5) are currently not used but are given a default value of zero and they may be used in a later release of the code.

2.2.2 Information arrays

The entries of the arrays INFO and RINFO provide information on the action of ME62A/AD, ME62J/JD, ME62P/PD, ME62B/BD, and ME62C/CD.

INFO(1) is used as an error flag. If a call to a routine in the ME62 package is successful, on exit INFO(1) has value 0. A negative value of INFO(1) indicates an error has been detected and a value greater than zero indicates a warning has been issued (see Section 2.3). If an error is detected during a call to ME62J/JD, the information contained in INFO(I), $3 \le I \le 6$, and RINFO(1) and RINFO(2) will be incomplete. Likewise, if an error is detected during a call to ME62B/BD, the information contained in INFO(I), $I \ge 9$, and in RINFO(I), $I \ge 3$, will be incomplete.

INFO(2) holds additional information concerning the error (see Section 2.3).

INFO(3) holds, on successful exit from the final call to ME62J/JD, the total number of variables in the problem.

INFO(4) holds, on successful exit from the final call to ME62J/JD, the number of static condensation variables (a static condensation variable is one which appears in only one element).

INFO(5) holds, on successful exit from the final call to ME62J/JD, the largest number of variables eliminated at a single stage (that is, the maximum order of a pivot block).

`INFO(6)` holds, on successful exit from the final call to `ME62J/JD`, the maximum front size.

`INFO(7)` and `INFO(8)` are currently not used but are initialised on the first call to `ME62J/JD` to zero and they may be used in a later release of the code.

`INFO(9)` holds, on successful exit from the final call to `ME62B/BD`, the number of pivots with negative real part.

`INFO(10)` holds, on successful exit from the final call to `ME62B/BD`, the number of buffers used to hold the entries in the factors.

`INFO(11)` holds, on successful exit from the final call to `ME62B/BD`, the number of buffers used to hold the indices of the variables in the factors.

`INFO(12)` holds, on successful exit from the final call to `ME62B/BD`, the maximum number of buffers required to hold the entries in a block of pivot rows.

`INFO(13)` holds, on successful exit from the final call to `ME62B/BD`, the maximum number of buffers required to hold the indices of the variables in a block of pivot rows.

`INFO(14)` to `INFO(20)` are currently not used but are initialised on the first call to `ME62B/BD` to zero and they may be used in a later release of the code.

`RINFO(1)` holds, on successful exit from the final call to `ME62J/JD`, the length in `COMPLEX` (`COMPLEX*16` in the D version) words of the file required by the numerical factorization to hold the entries in the factors (no allowance is made for right-hand sides or for possible exploitation of zeros in the front).

`RINFO(2)` holds, on successful exit from the final call to `ME62J/JD`, the length in `INTEGER` words of the file required by the numerical factorization to hold the indices of the variables in the factors (no allowance is made for possible exploitation of zeros in the front).

`RINFO(3)` holds, on successful exit from the final call to `ME62B/BD`, the length in `COMPLEX` (`COMPLEX*16` in the D version) words of the file actually used during the factorization to hold the entries in the factors and the corresponding right-hand sides. If `NRHSB` = `0` and `ICNTL(9)` = `0`, `RINFO(3)` = `RINFO(1)`.

`RINFO(4)` holds, on successful exit from the final call to `ME62B/BD`, the number of entries in the factors that have value zero. If there a large number of zeros in the factors and zeros in the front have not been exploited (`ICNTL(9)` = `0`), the user should try a smaller value of `ICNTL(5)`, or reset `ICNTL(9)` to a nonzero value, or reorder the elements.

`RINFO(5)` holds, on successful exit from the final call to `ME62B/BD`, the number of entries (including zero entries) stored in the factors. If zeros in the front are not exploited (`ICNTL(9)` = `0`), `RINFO(5)` = `RINFO(1)`.

`RINFO(6)` holds, on successful exit from the final call to `ME62B/BD`, the number of integers actually used during the factorization to store the factors. If zeros in the front are not exploited (`ICNTL(9)` = `0`), `RINFO(6)` = `RINFO(2)`.

`RINFO(7)` holds, on successful exit from the final call to `ME62B/BD`, the natural logarithm of the modulus of the determinant of the matrix **A**.

`RINFO(8)` holds, on successful exit from the final call to `ME62B/BD`, the number of floating-point operations to perform the factorization. This count includes operations performed during static condensation.

`RINFO(9)` holds, on successful exit from the final call to `ME62B/BD`, the root-mean-squared front size.

`RINFO(10)` to `RINFO(20)` are currently not used but are initialised on the first call to `ME62B/BD` to zero and they may be used in a later release of the code.

2.3 Error diagnostics

On successful completion, the subroutines in the `ME62` package will exit with the parameter `INFO(1)` set to `0`. Other values for `INFO(1)` and the reasons for them are given below.

A negative value for `INFO(1)` is associated with a fatal error. If `ICNTL(8)` > `0` and `ICNTL(1)` > `0`, a self-explanatory message is, in each case, output on unit ICNTL(1) (see Section 2.2.1). The negative values for INFO(1) are:

–1 LENLST ≤ 0 on entry to ME62A/AD. (ME62A/AD first entry only).

–2 NVAR ≤ 0 in the current element. (ME62A/AD, ME62J/JD, and ME62B/BD entries). This error is also returned if NVAR is greater than LAVAR (ME62B/BD entries only).

–3 An index of a variable in the current element is out of range. INFO(2) holds the index which is out of range. (ME62A/AD, ME62J/JD, and ME62B/BD entries).

–4 Duplicate occurrences of the same variable index found in the current element. INFO(2) holds the duplicated index. (ME62A/AD entries only).

–5 NRHSB ≥ 1 and the defined first dimension LX of the array X is less than NDF as output from the final call to ME62A/AD. INFO(2) holds the value of NDF output from ME62A/AD. (ME62B/BD first entry only). This error is also returned by ME62C/CD if the defined first dimension LX of the array X is less than NDF.

–6 Defined length LW of the real workspace array W violates the restrictions on LW. LW must be increased to at least INFO(2). (ME62B/BD first entry only and ME62C/CD entry).

–7 Defined length LIW of the integer workspace array IW violates the restrictions on LIW. LIW must be increased to at least INFO(2). (ME62B/BD first entry only and ME62C/CD entry).

–8 A call to ME62J/JD has not been preceded by calls to ME62A/AD. (ME62J/JD first entry only).

–9 The number of right-hand sides is out of range. Either NRHSB < 0 (ME62B/BD first entry only) or the user has changed the number of right-hand sides between calls to ME62B/BD. INFO(2) holds the value of NRHSB on the first call to ME62B/BD. This error is also returned by ME62C/CD if NRHSC < 1.

–10 Either the order of the elements or the index list for one or more of the elements has been changed since the calls to ME62J/JD. (ME62B/BD entries only).

–11 A variable appears again after it has been fully summed (this happens if an index list for an element has been changed since ME62A/AD was called, or the order of the elements has been changed, or more elements have been entered than were entered to ME62A/AD). INFO(2) holds the index of the fully summed variable. (ME62J/JD and ME62B/BD entries).

–12 Attempt to use a pivot of absolute value less than or equal to CNTL(1). INFO(2) holds the call on which this error was encountered. (ME62B/BD entries only).

–13 The value of NDF has been changed since the final call to ME62A/AD. INFO(2) holds the value of NDF output from ME62A/AD. (ME62J/JD and ME62B/BD first entries only).

–14 The number of calls made to ME62J/JD differs from the number of calls made to ME62A/AD. This error is also returned if ME62B/BD is called without ME62J/JD being called. INFO(2) holds the number of calls made to ME62J/JD. (ME62B/BD first entry only).

–15 LENBUF(1) or LENBUF(2) violates the restrictions on it.
LENBUF(I) ≤ 0, I = 1 or 2 (ME62P/PD entry only), or
LENBUF(1) < RINFO(1) + NDF*NRHSB, or LENBUF(2) < RINFO(2), (ME62B/BD first entry only, direct access files not in use), or
LENBUF(I), I = 1 or 2, has been changed between the call to ME62P/PD and the first call to ME62B/BD.

–16 ISTRM(1)=ISTRM(2) or ISTRM(I) lies out of range, or is equal to 6, ICNTL(1), or ICNTL(2) (I = 1 or 2). (ME62P/PD entry only).

–17 Error encountered in Fortran OPEN statement. INFO(2) holds the IOSTAT parameter (the IOSTAT parameter is a parameter which, after an input/output operation is completed, is set to zero if no error was detected and to a positive integer otherwise). (ME62P/PD entry only).

–18 ICNTL(I) ≤ 0 for I = 3 or 4. (ME62P/PD entry only).

–19 Error detected when reading a direct access file. INFO(2) holds the IOSTAT parameter. (ME62B/BD and ME62C/CD entries).

–20 Error detected when writing to a direct access file. INFO(2) holds the IOSTAT parameter (ME62B/BD entries only).

–22 The user has changed KIND between calls to ME62B/BD or between the final call to ME62B/BD and a call to ME62C/CD (ME62B/BD and ME62C/CD entries only) .

Warning messages are associated with a positive value for INFO(1). If ICNTL(8) > 1 and ICNTL(2) > 0, a self-explanatory message is, in each case, output on unit ICNTL(2) (see Section 2.2.1). The warnings are:

+1 On entry to ME62J/JD, ICNTL(5) is less than or equal to zero. The default value 16 is used. (ME62J/JD first entry only).

+2 On entry to ME62B/BD, ICNTL(5) is not equal to the value used by ME62J/JD. The value used by ME62J/JD is used. (ME62B/BD first entry only).

+4 On entry to ME62B/BD or ME62C/CD, ICNTL(1) (or ICNTL(2)) has a value equal to ISTRM(1) or ISTRM(2). The default value 6 is used. (ME62B/BD first entry only and ME62C/CD entry). This warning can only be issued if ME62P/PD has been called.

+x x = 3, 5, 6, 7 are combinations of the above warnings corresponding to summing the constituent warnings.

3 GENERAL INFORMATION

Use of common: None.

Other routines called directly: The BLAS routines CAXPY/ZAXPY, CGERU/ZGERU, CGEMV/ZGEMV, CTPSV/ZTPSV, CTRSV/ZTRSV, CGEMM/ZGEMM, CTRSM/ZTRSM. Subroutines internal to the package are ME62D/DD, ME62E/ED, ME62F/FD, ME62G/GD, ME62H/HD, ME62L/LD, ME62M/MD, ME62N/ND, ME62O/OD.

Workspace: Workspace is provided by the arrays:
W(LW) (ME62B/BD and ME62C/CD).
IW(LIW) (ME62B/BD, and ME62C/CD).
LAST(NDF) is used locally as workspace (ME62B/BD only).
ISAVE(50) is a work array that must be unchanged between calls to routines in the ME62 package.

Input/output: In the event of errors, diagnostic messages are printed. The output streams for these messages are controlled by the variables ICNTL(1) and ICNTL(2), and the level of printing is controlled by ICNTL(8) (see Section 2.2.1). Stream ICNTL(1) is used for error messages (INFO(1) < 0) and stream ICNTL(2) for warnings (INFO(1) > 0) and diagnostic printing.

Restrictions:

ME62A/AD:

NVAR ≥ 1.

LENLST ≥ 1.

1 ≤ IVAR(I) ≤ LENLST and IVAR(I) ≠ IVAR(J), I, J, = 1, 2, ..., NVAR.

ME62J/JD:

NVAR ≥ 1.

1 ≤ IVAR(I) ≤ NDF, I = 1, 2, ..., NVAR.

ME62P/PD:

ISTRM(1) and ISTRM(2) lie in the range [1, 99] and do not equal 6, ICNTL(1), or ICNTL(2).

ISTRM(1) ≠ ISTRM(2).

ICNTL(I) > 0, I = 3, 4.

LENBUF(I) > 0, I = 1, 2.

ME62B/BD:

NVAR ≥ 1.

1 ≤ IVAR(I) ≤ NDF, I = 1, 2, ..., NVAR.

LAVAR ≥ NVAR.

NRHSB ≥ 0.

If NRHSB ≥ 1, LX ≥ NDF.

If ME62P/PD is not called, LENBUF(1) ≥ RINFO(1) + NDF*NRHSB, LENBUF(2) ≥ RINFO(2).

LW ≥ LENBUF(1) + INFO(6)*(INFO(6) + NRHSB) + 3

LIW ≥ LENBUF(2) + 3*INFO(6).

ME62C/CD:

NRHSC ≥ 1.

LX ≥ NDF.

If ME62P/PD is not called,
LW ≥ LENBUF(1) + INFO(6)*NRHSC + 3
LIW ≥ LENBUF(2) + INFO(6) + 4
otherwise,
LW ≥ LENBUF(1) + INFO(6)*NRHSC + INFO(5)*(INFO(6)+NRHSB).
LIW ≥ LENBUF(2).

4 METHOD

The method used is a modification of the unsymmetric frontal code of Duff and Scott (1993, 1996) and is a complex version of MA62 (Duff and Scott 1997).

The elements are assembled into an in-core frontal matrix one at a time. A variable which has appeared for the last time (i.e. does not occur in future elements) is fully summed and is used as a pivot in Gaussian elimination, provided it is of absolute value at least CNTL(1) and there are at least ICNTL(5) fully summed variables. Once all possible eliminations for the current element have been performed, the pivot columns are written to in-core buffers and later, if requested, to direct access files. To prevent the amount of in-core memory required becoming too large, the user should order the elements so that the same variable does not occur in elements that are widely apart in the ordering. Thus, for example, in a problem with a narrow pipe geometry, the elements should be ordered across the cross-section of the pipe rather than along its length. An efficient element ordering can be obtained using the HSL routine MC63.

References.

Duff, I.S. and Scott, J.A. (1993). MA42 – A new frontal code for solving sparse unsymmetric systems. Report RAL-93-064, Rutherford Appleton Laboratory.

Duff, I.S. and Scott, J.A. (1996). The design of a new frontal code for solving sparse unsymmetric systems. *ACM Trans. Math. Softw.*, **22**, 30-45.

Duff, I.S. and Scott, J.A. (1997). MA62 – A frontal code for sparse positive-definite symmetric systems. Technical Report RAL-TR-97-012, Rutherford Appleton Laboratory. To appear in *ACM Trans. Math. Softw.*.

5 EXAMPLE OF USE

We give an example of the code required to solve a set of complex symmetric finite-element equations using the ME62 package. The example illustrates the full calling sequence for the ME62 package. In this example, we wish to solve $\mathbf{AX}=\mathbf{B}$. We supply one right-hand side with the elements and then use ME62C/CD to solve for a further two right-hand sides. Direct access files are used to hold the factors.

Consider a 5×5 complex symmetric matrix

$$\begin{pmatrix} 3+6i & 0 & 0 & 0 & 2 \\ 0 & 5+2i & 3+3i & 6 & 0 \\ 0 & 3+3i & 3+8i & 6+6i & 0 \\ 0 & 6 & 6+6i & 9+7i & 2+i \\ 2 & 0 & 0 & 2+i & 5+5i \end{pmatrix}$$

generated from four elemental matrices

$$\begin{matrix}4\\5\end{matrix}\begin{pmatrix} 4+3i & 2+i \\ 2+i & 4+4i \end{pmatrix} \qquad \begin{matrix}1\\5\end{matrix}\begin{pmatrix} 3+6i & 2 \\ 2 & 1+i \end{pmatrix} \qquad \begin{matrix}2\\3\\4\end{matrix}\begin{pmatrix} 5+2i & 3+3i & 6 \\ 3+3i & 1+5i & 1+i \\ 6 & 1+i & 2+4i \end{pmatrix} \qquad \begin{matrix}3\\4\end{matrix}\begin{pmatrix} 2+3i & 5+5i \\ 5+5i & 3 \end{pmatrix},$$

where the variable indices are indicated by the integers before each matrix (columns are identified symmetrically to rows). The corresponding elemental vectors $\mathbf{b}^{(k)}$ $(1\le k\le 4)$ are

$$\begin{pmatrix} 2+10i \\ 1+11i \end{pmatrix} \qquad \begin{pmatrix} -1+11i \\ 2+4i \end{pmatrix} \qquad \begin{pmatrix} 9+19i \\ -4+14i \\ 4+14i \end{pmatrix} \qquad \begin{pmatrix} -1+15i \\ 3+13i \end{pmatrix}.$$

The (assembled) right-hand sides that are used in the call to ME62C/CD are

$$\begin{pmatrix} -1+11i \\ 4+10i \\ -10+12i \\ 1+15i \\ 2+12i \end{pmatrix} \qquad \begin{pmatrix} 0 \\ 20-4i \\ 18-4i \\ 28-14i \\ 3-i \end{pmatrix}.$$

The following program is used to solve this problem. In this program, we read the element data into arrays ELTPTR (location of first entry of element), ELTVAR (variable indices), VALUE (numerical values), and RHSVAL (right-hand sides). This method of storing the element data is used here for illustrative purposes; the user may prefer, for example, to read in the element data from a direct access file.

```
C Example to illustrate the use of ME62.
C     .. Parameters ..
      INTEGER MAXE,LIWMAX,LRHS,NZMAX,MELT,LENLST,LWMAX,MAXVL,MAXRVL,
     +        NDFMAX
      PARAMETER (MAXE=4,LIWMAX=120,LRHS=2,NZMAX=15,MELT=4,LENLST=5,
     +          LWMAX=120,MAXVL=30,MAXRVL=10,NDFMAX=5)
C     ..
C     .. Local Scalars ..
      INTEGER I,IELT,J,JSTRT,K,KSTRT,LAVAR,LIW,LW,LX,NDF,
     +        NELT,NFRONT,NPIV,NRHSB,NRHSC,NVAR,NZ,RHSCRD,VALCRD
      LOGICAL KIND
C     ..
C     .. Local Arrays ..
      COMPLEX*16 AVAR(MAXE,MAXE),RHS(MAXE,LRHS),RHSVAL(MAXRVL),
     +           VALUE(MAXVL),W(LWMAX),X(NDFMAX,LRHS)
      DOUBLE PRECISION CNTL(5),RINFO(20)
      INTEGER ELTPTR(MELT+1),ELTVAR(NZMAX),ICNTL(15),
```

```
     +        INFO(20),ISAVE(50),ISTRM(2),IVAR(MAXE),IW(LIWMAX),
     +        LAST(LENLST),LENBUF(2)
      CHARACTER FILNAM(2)*128
C     ..
C     .. External Subroutines ..
      EXTERNAL ME62AD,ME62BD,ME62CD,ME62ID,ME62JD,ME62PD
C     ..
C*** Call to ME62ID
      CALL ME62ID(ICNTL,CNTL,ISAVE)

C Read in the element data.
C NELT is number of elements.
C ELTVAR contains lists of the variables belonging to the finite
C elements, with those for element 1 preceding those for element
C 2, and so on. ELTPTR points to the position in ELTVAR
C of the first variable in element I. NZ is the total number
C of entries in the element lists.

      READ (5,FMT=*) NELT
      READ (5,FMT=*) (ELTPTR(I),I=1,NELT+1)
      NZ = ELTPTR(NELT+1) - 1
      READ (5,FMT=*) (ELTVAR(I),I=1,NZ)

C Calls to ME62AD to establish when each variable is fully assembled
      DO 20 IELT = 1,NELT
         NVAR = ELTPTR(IELT+1) - ELTPTR(IELT)
         JSTRT = ELTPTR(IELT)
         DO 10 J = 1,NVAR
            IVAR(J) = ELTVAR(JSTRT+J-1)
   10    CONTINUE
C*** Call to ME62AD
         CALL ME62AD(NVAR,IVAR,NDF,LAST,LENLST,ICNTL,ISAVE,INFO)
         IF (INFO(1).LT.0) GO TO 100
   20 CONTINUE

C Calls to ME62JD to determine file sizes
      DO 40 IELT = 1,NELT
         NVAR = ELTPTR(IELT+1) - ELTPTR(IELT)
         JSTRT = ELTPTR(IELT)
         DO 30 J = 1,NVAR
            IVAR(J) = ELTVAR(JSTRT+J-1)
   30    CONTINUE
C*** Call to ME62JD
         CALL ME62JD(NVAR,IVAR,NDF,LAST,ICNTL,ISAVE,INFO,RINFO)
         IF (INFO(1).LT.0) GO TO 100
   40 CONTINUE
      WRITE (6,FMT='(/A)') ' On exit from ME62JD:'
      WRITE (6,FMT='(/A/6I5)') ' INFO(1:6)    = ',(INFO(I),I=1,6)
      WRITE (6,FMT='(/A/2(F10.3))') ' RINFO(1:2)    = ',(RINFO(I),I=1,2)

C Call to ME62PD to open direct access data sets.
C Choose stream numbers and file sizes.
      ISTRM(1) = 8
      ISTRM(2) = 9
      LENBUF(1) = 30
      LENBUF(2) = 30
C*** Call to ME62PD
      CALL ME62PD(ISTRM,FILNAM,LENBUF,ICNTL,ISAVE,INFO)
      IF (INFO(1).LT.0) GO TO 100

C Input elemental matrices and right-hand sides.
```

```
C VALCRD is the number of numerical values to be input.
C VALUE contains lists of the numerical values in the elemental
C matrices, with element 1 preceding element 2, and so on.
C Since the elemental matrices are symmetric only the upper
C triangular part is stored.
      KIND = .FALSE.
      READ (5,FMT=*) VALCRD
      READ (5,FMT=*) (VALUE(I),I=1,VALCRD)

C RHSCRD is the number of right-hand side numerical values to
C be input.
C RHSVAL contains lists of the right-hand side numerical values
C corresponding to each of the elements in order.

      READ (5,FMT=*) RHSCRD
      READ (5,FMT=*) (RHSVAL(I),I=1,RHSCRD)
C
C Prepare to call ME62BD.
      LAVAR = MAXE
      NRHSB = 1
      LX = NDFMAX
      NFRONT = INFO(6)
      LW = 3 + LENBUF(1) + NFRONT* (NFRONT+NRHSB)
      LIW = LENBUF(2) + 3*NFRONT
      IF (LW.GT.LWMAX .OR. LIW.GT.LIWMAX) GO TO 110
C
      KSTRT = 1
      DO 70 IELT = 1,NELT
         NVAR = ELTPTR(IELT+1) - ELTPTR(IELT)
         JSTRT = ELTPTR(IELT)
         DO 60 J = 1,NVAR
            IVAR(J) = ELTVAR(JSTRT+J-1)
            DO 50 K = J,NVAR
               AVAR(J,K) = VALUE(KSTRT)
               KSTRT = KSTRT + 1
   50       CONTINUE
            RHS(J,1) = RHSVAL(JSTRT+J-1)
   60    CONTINUE
C*** Call to ME62BD
         CALL ME62BD(KIND,NVAR,IVAR,NDF,LAST,LAVAR,AVAR,NRHSB,RHS,
     +               LX,X,LENBUF,LW,W,LIW,IW,ICNTL,CNTL,ISAVE,
     +               INFO,RINFO)
         IF (INFO(1).LT.0) GO TO 100
   70 CONTINUE

C  Solution is in first NDF locations of X
      WRITE (6,FMT='(/A)') ' On exit from ME62BD:'
      WRITE (6,FMT='(/A)') ' The solution is:'
      WRITE (6,FMT='(2(F10.2))') (X(I,1),I=1,NDF)
      WRITE (6,FMT='(/A/5I5)') ' INFO(9:13)    = ',(INFO(I),I=9,13)
      WRITE (6,FMT='(/A/7(F10.3))') ' RINFO(3:9)    = ',(RINFO(I),I=3,9)

C Now read in further right-hand sides

      READ (5,FMT=*) NRHSC
      DO 80 J = 1,NRHSC
         READ (5,FMT=*) (X(I,J),I=1,NDF)
   80 CONTINUE
      NPIV = INFO(5)
      LW = LENBUF(1) + NFRONT*NRHSC + NPIV*(NFRONT+NRHSC)
      LIW = LENBUF(2) + NFRONT + 4
```

```
      IF (LW.GT.LWMAX .OR. LIW.GT.LIWMAX) GO TO 100
C*** Call to ME62CD
      CALL ME62CD(KIND,NRHSC,NDFMAX,X,LW,W,LIW,IW,ICNTL,ISAVE,INFO)
      IF (INFO(1).LT.0) GO TO 100

C  Solution for J-th right-hand side is in X(.,J), J=1,NRHS
      WRITE (6,FMT='(/A)') ' On exit from ME62CD:'
      DO 90 J = 1,NRHSC
         WRITE (6,FMT='(/3X,A,I2,A)')
     +   'The solution for right hand side number',J,' is:'
         WRITE (6,FMT='(2(F10.2))') (X(I,1),I=1,NDF)
   90 CONTINUE
      GO TO 110
C Print appropriate fatal error diagnostic
  100 WRITE (6,FMT='(/A,I3)') 'Error return. INFO(1) = ',INFO(1)
  110 STOP
      END
```

The input data used for this problem is:

```
4
1   3   5   8  10
4   5   1   5   2   3   4   3   4
15
(4.,  3.)  (2.,  1.)  (4.,  4.)  (3.,  6.)  (2.,  0.)
(1.,  1.)  (5.,  2.)  (3.,  3.)  (6.,  0.)  (1.,  5.)
(1.,  1.)  (2.,  4.)  (2.,  3.)  (5.,  5.)  (3.,  0.)
9
(2.,  10.)  (1.,  11.)  (-1., 11.)  (2.,   4.)  (9.,  19.)
(-4., 14.)  (4.,  14.)  (-1., 15.)  (3.,  13.)
2
(-1., 11)  (4.,  10),  (-10., 12.)  (1.,   15.)  (2.,  12.)
(0.,   0)  (20., -4),  (18.,  -4.)  (28., -14.)  (3.,  -1.)
```

This produces the following output:

```
On exit from ME62JD:

INFO(1:6)    =
   0    0    5    2    3    3

RINFO(1:2)    =
   11.000    21.000

On exit from ME62BD:

The solution is:

     1.00      1.00
     1.00      1.00
     1.00      1.00
     1.00      1.00
     1.00      1.00

INFO(1,2,9:13)    =
   0    0    0    1    1    1    1

RINFO(3:9)    =
   16.000     1.000    11.000    21.000
    9.009    80.000     2.793

On exit from ME62CD:

  The solution for right hand side number 1 is:

     1.00      1.00
     2.00      0.00
     0.00      2.00
     0.00      0.00
     1.00      1.00

  The solution for right hand side number 2 is:

     0.00      0.00
     4.00      0.00
    -2.00      0.00
     1.00     -1.00
     0.00      0.00
```